1. Introduction

1.1 The Goal of this Primer

Our objective is to introduce you to the fundamentals of seismic data processing with a learn-by-doing approach. We do this with Seismic Un*x (SU), a free software package maintained and distributed by the Center for Wave Phenomena (CWP) at the Colorado School of Mines (CSM). At the outset, we want to express our gratitude to John Stockwell of the CWP for his expert counsel.

SU runs on several operating systems, including Unix, Microsoft Windows, and Apple Macintosh. However, we discuss SU only on Unix.

Detailed discussion of wave propagation, convolution, cross- and auto-correlation, Fourier transforms, semblance, and migration are too advanced for this Primer. Instead, we suggest you refer to other publications of the Society of Exploration Geophysicists, such as "Digital Processing of Geophysical Data - A Review" by Roy O. Lindseth and one of the two books by Ozdogan Yilmaz: "Seismic Data Processing," 1987 and "Seismic Data Analysis," 2001.

Our goal is to give you the experience and tools to continue exploring the concepts of seismic data processing on your own.

1.2 The Outline of this Primer

This Primer covers all processing steps necessary to produce a time migrated section from a 2-D seismic line. We use three sources of input data:

- Synthetic data generated by SU;
- Real shot gathers from the Oz Yilmaz collection at the Colorado School of Mines (ftp://ftp.cwp.mines.edu/pub/data); and
- Real 2-D marine lines provided courtesy of Prof. Greg Moore of the University of Hawaii: the "Nankai" data set and the "Taiwan" data set.
    - The University of Texas, the University of Tulsa, and the University of Tokyo collected the Nankai data. The U.S. National Science Foundation and the government of Japan funded acquisition of the Nankai data.
    - The University of Hawaii, San Jose State University, and National Taiwan University collected the Taiwan data. The U.S. National Science Foundation and the National Science Council of Taiwan funded acquisition of the Taiwan data.

Chapters 1-3 introduce the Unix system and Seismic Un*x.

Chapters 4-5 build three simple models (complexity slowly increases) and acquire a 2-D line over each model. (These chapters may be skipped if you are only interested in processing.)

Chapters 6-9 build a model based on the previous three, acquire a 2-D line over that model, and process the line through migration.

Chapters 10-11 start with a real 2-D seismic line of shot gathers (Nankai) and process it through migration.

Downloaded 01/21/14 to 132.178.2.64. Redistribution subject to SEG license or copyright; see Terms of Use at http://library.seg.org/

Chapters 12-13 and 15-16 start with a real 2-D line of shot gathers (Taiwan) and process it through migration.

Chapter 14 uses real shot gathers from the Oz Yilmaz collection to demonstrate f-k filtering and deconvolution.

We introduce the major processing steps in the following chapters:

Build a model	Chapter 4
Acquire seismic data	Chapter 5
Amplitude correction	Chapter 10
Frequency (1-D) filtering	Chapter 10
Frequency-wavenumber (*f-k*) filtering	Chapter 14
Deconvolution	Chapter 14
Sort from shot gathers to CMP gathers	Chapter 7
Velocity analysis	Chapter 7
Normal moveout correction (NMO)	Chapter 8
Stack	Chapter 8
Migration	Chapter 9

Scripts are included for each process. A detailed description of these scripts is the heart of this Primer.

Experience with the Unix operating system is helpful, but not necessary. Chapter 2 introduces Unix commands and SU scripts.

1.3 A Bit about Seismic Un*x

Seismic Un*x (Cohen & Stockwell, 2002) is a free software package for seismic data processing. SU is maintained and regularly updated by the Center for Wave Phenomena (CWP) at the Colorado School of Mines (CSM). The SU home page at the CWP is:

http://www.cwp.mines.edu/cwpcodes/

You can download SU from the CWP ftp server and install it on almost any Unix system. The source code is also included, making it possible for you to adjust and expand SU. The CWP web site also contains links to other useful web sites.

Please read the legal statement that accompanies SU. For your convenience, we make this available as our Appendix A.

SU is easy to use because it does not require extensive programming knowledge, only basic Unix commands for file management, pipes, and shell redirection (see Chapter 2).

Although SU can be run from the command line, the most efficient way to use SU is through shell scripts. Shell scripts let us combine several processing programs into one "job," similar to commercial seismic processing packages. We first describe Shell scripts in Chapter 2.

1.4 Computer Note 1: Enter

When we tell you to “enter” something on the computer, we want you to type the instruction, then press the **Enter** or the **Return** key. Pressing **Enter** or **Return** tells the computer to execute the instruction. Merely typing an instruction on a line does not execute the instruction.

Downloaded 01/21/14 to 132.178.2.64. Redistribution subject to SEG license or copyright; see Terms of Use at http://library.seg.org/

Downloaded 01/21/14 to 132.178.2.64. Redistribution subject to SEG license or copyright; see Terms of Use at http://library.seg.org/

1.5 Computer Note 2: Your Shell

When you enter a command, you are interacting with the Unix shell. There are several Unix shells: the Bourne shell, the C shell, the Korn shell, the Bourne Again shell (BASH), and others. **All example scripts in this Primer use the Bourne shell; most of the Computer Notes refer to the C shell.**

1.6 Computer Note 3: Changing Directories

1. Open an x-term window and enter **`pwd`** (on some computers you have to use **cwd**) to see the directory in which you are now working.
2. Enter **`ls`** to see a listing of the files and subdirectories in that directory.
3. If you have a subdirectory; for example, "work," enter **`cd work`** to change to that subdirectory, then enter **`ls`** to see a listing within it.
4. When you want to move up one level of directory, enter **`cd ..`** (Yes, that is: `cd space dot dot.`)
5. If you enter **`cd`** by itself, you will go to your home directory.
6. If you enter **`cd .`** (`cd space dot`), you will not go anywhere because the single dot (period) is a reference to your current directory.

   The single dot can be useful. Suppose there is a system calculator called xcalc and, suppose you wrote your own calculator program that is also called xcalc. If you enter

   ```
   xcalc
   ```

   the system version might start instead of yours. To run your version instead of the system version, go to the directory that contains your version, then enter

   ```
   ./xcalc
   ```

   The dot slash tells the system to look in the current directory for the program.

1.7 Computer Note 4: Cursor Prompt

Usually, the default computer prompt does not give much information. For example, Figure 1.1 shows the default prompt when I (Forel) open a new x-term window

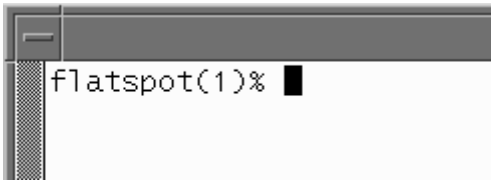

Figure 1.1: *Forel's default x-term prompt.*

The prompt of Figure 1.1 shows the name of the machine that I am on and the number of the command (1) that I am about to enter. I think a more useful prompt is one that tells me which directory I am in (Figure 1.2).

A disadvantage of a prompt that shows the current directory is that as I go to deeper subdirectories, the prompt becomes longer (Figure 1.3). I am willing to pay this price for the information that I gain.

Figure 1.2: *My new prompt shows the current directory I also changed "%' to ">".*

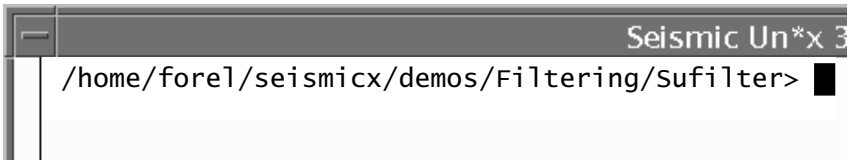

Figure 1.3: *A prompt with a long path name.*

To change your C shell prompt, edit your home directory's shell resource file. The file ".cshrc" is the resource file for the C shell. Yes, the file name starts with a period (.). Use any editor you like to add the following lines to your ".cshrc" file.

Note: We use only Unix-based text editors to write and edit shell scripts and resource files. Many personal computer editors insert hidden format codes that might be wrongly interpreted by the Unix operating system.

```
# Make prompt contain "pwd"
alias setprompt 'set prompt="$cwd> "'
alias cd 'cd \!* && setprompt'
alias pushd 'pushd \!* && setprompt'
alias popd 'popd \!* && setprompt'
setprompt
```

The first line is a comment. When the first character is "#," the entire line is ignored.

After you save your changes, when you open a new x-term window, the system will read the resource file and use those instructions in the new x-term window. If you want to use .cshrc changes in your current window, enter:

```
source  .cshrc
```

1.8 Your First Seismic Un*x Instruction

Before you can run Seismic Un*x, you must be sure your SU commands will be interpreted by the Unix operating system. This means SU must be "in your path." Your system administrator may have made SU part of your login path or you may have to enter a command first. The approach we use at Michigan Tech is described in Appendix B.

Let's run a simple program to test that you and your SU environment are working together. Enter (type the following, then press the **Enter** or the **Return** key):

```
suplane | suxwigb &
```

You should get a new window with a picture of some synthetic seismic data on your screen (Figure 1.4). If you do not, ask someone for help.

The style of the above command is:

```
[program]  ["pipe" output data to next program]  [program]  &
```

The pipe "|" directs data from one process to another.

Downloaded 01/21/14 to 132.178.2.64. Redistribution subject to SEG license or copyright; see Terms of Use at http://library.seg.org/

You can regain the Unix prompt in your SeismicX window by pressing the return key (if you supplied **&** at the end of the command). You can close the graphics window by using the Unix **Close** command (the top left button).

Let's discuss the details of the above command:

- **suplane** creates seismic data.
- the pipe (the vertical bar) directs (sends) the data to the next program.
- **suxwigb** (the X-window display program) displays the seismic data on your screen as wiggle traces.
- The **&** needs a special explanation. Read the next section.

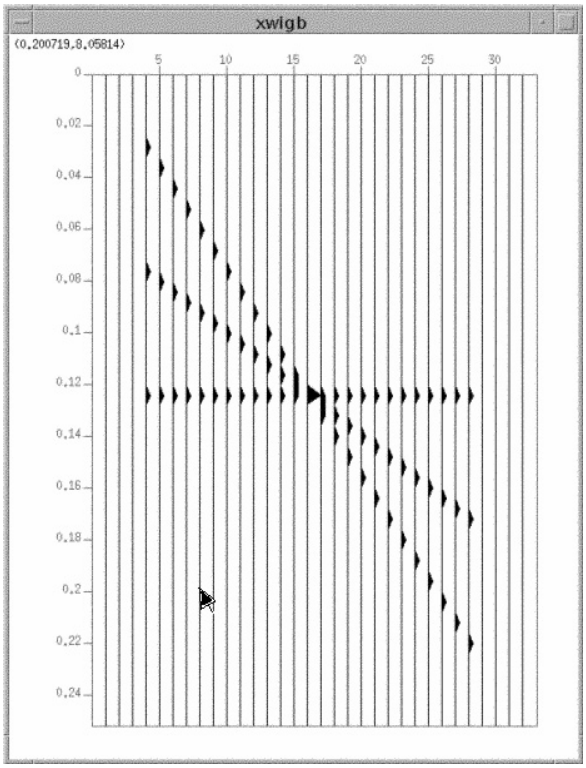

Figure 1.4: *The output of your first SU command. The upper-left corner shows (0.200719,8.05814) because we clicked the middle mouse button at approximately 0.200 ms time (the vertical axis is time) and on the eighth trace from the left (the horizontal axis is trace number).*

1.9 Computer Note 5: Background (&) and Foreground Processes

1.9.1 Background Processes

In technical language, using **&** (ampersand) at the end of a command runs (spawns) the process in the background. In simple terms, using **&** does not lock your cursor. Since your cursor is not locked (frozen) when you use **&** at the end of a command, you can use the same x-term to do other things, like submit other processes. For example, the following command starts the Netscape web browser as a background process:

```
netscape  &
```

To enter other commands after this one, press Enter again to regain a prompt, then enter your next command.

Downloaded 01/21/14 to 132.178.2.64. Redistribution subject to SEG license or copyright; see Terms of Use at http://library.seg.org/

Downloaded 01/21/14 to 132.178.2.64. Redistribution subject to SEG license or copyright; see Terms of Use at http://library.seg.org/

While a process is running in the background, you might need to cancel (kill) it. You can do this using a Unix utility called **top**.

1. In an x-term window, enter **top**
2. The x-term window now shows your most active processes (Figure 1.5) and the cursor is now in the upper left of the **top** display (shown as the red box in Figure 1.5).

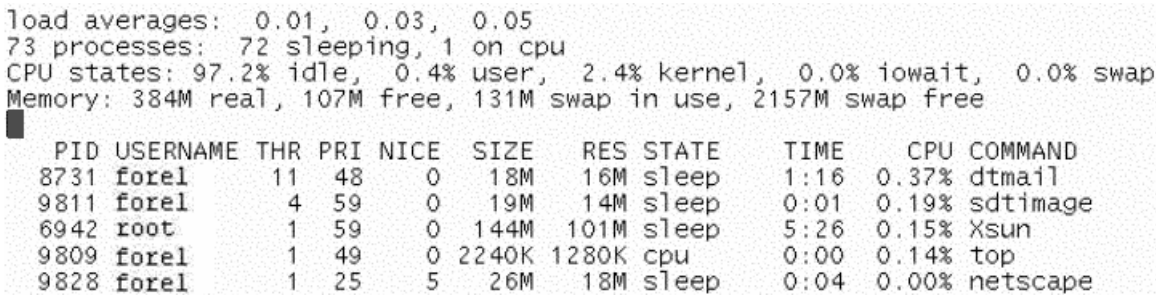
load averages: 0.01, 0.03, 0.05
73 processes: 72 sleeping, 1 on cpu
CPU states: 97.2% idle, 0.4% user, 2.4% kernel, 0.0% iowait, 0.0% swap
Memory: 384M real, 107M free, 131M swap in use, 2157M swap free
PID USERNAME THR PRI NICE SIZE RES STATE TIME CPU COMMAND
8731 forel 11 48 0 18M 16M sleep 1:16 0.37% dtmail
9811 forel 4 59 0 19M 14M sleep 0:01 0.19% sdtimage
6942 root 1 59 0 144M 101M sleep 5:26 0.15% Xsun
9809 forel 1 49 0 2240K 1280K cpu 0:00 0.14% top
9828 forel 1 25 5 26M 18M sleep 0:04 0.00% netscape

Figure 1.5: *Screen capture of Forel's* **top** *processes.*

3. The left column (PID) identifies process netscape, the Netscape web browser. as 9828. To kill this process, enter

```
k 9828
```

4. To exit **top**, enter **q** for quit.

Note: You can use **top** to learn the process ID (PID), then kill the process using a line command. To kill the Netscape process described above using a line command, enter:

```
kill -9 9828
```

The flag "-9" tells **kill** that you are serious about killing process 9828.

1.9.2 Foreground Processes

When you start a process and don't use **&**, the new process uses the current x-term (the process runs in the foreground), which locks the cursor. For example, the following command starts the Netscape web browser as a foreground process:

```
netscape
```

To continue working when the cursor is locked, you must either kill the process that you just started or you must open a new x-term. To kill a foreground process in an x-term window:

1. click the window to make it active,
2. hold down the **Control** key and briefly press the **c** key (press **Control-c**).

This kills the process. Because the process dies, the browser window also closes.

1.10 Zoom the xwigb Window

You can zoom the image in the **xwigb** window. Put the cursor somewhere in the **xwigb** window, press and hold down the left mouse button, drag the mouse until the box encloses the area you want enlarged, then let go of the button.

You can zoom repeatedly. Notice that time numbers and trace numbers correspond to the zoomed data. Click once in the window to return to the original view.

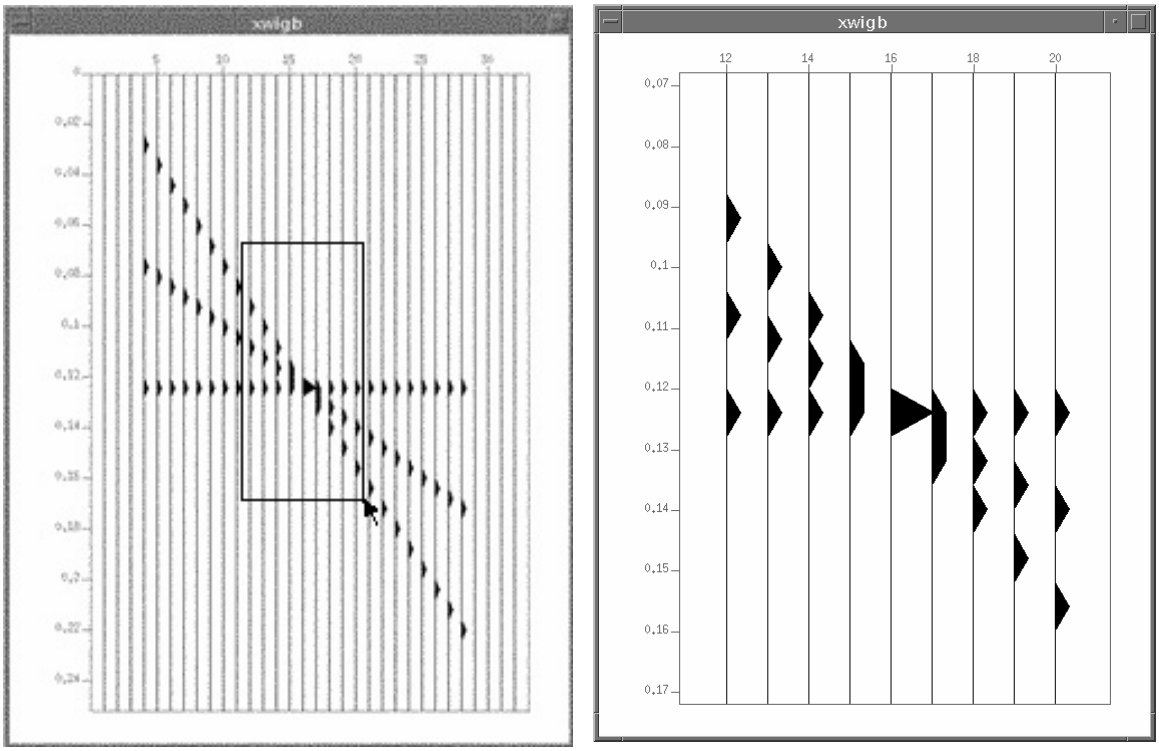

Figure 1.6: *Zoom an **xwigb** window by dragging a box with the left mouse button. The left image shows the mouse dragging the box. The right image shows the zoomed image. To return to the original view, click the left mouse button in the window.*

1.11 The Demos Directory

The **demos** directory is a standard part of the SU installation. If you don't know where yours is, ask someone. It is full of easy to use demonstrations of many seismic data processing programs.

Once you are in the **demos** directory, repeatedly use **cd** to move around. Use **ls** and **pwd** frequently to see where you are. Notice the many README files. The README files tell how to use the demonstrations in each directory.You can write README files to your screen by typing either of the following commands:

```
cat  README
more  README
```

If you use the second command, you can advance the text that appears on your screen by pressing the spacebar to move to the next page. Pressing **Enter** or **Return** scrolls the display one line.

Exploring (executing) the demonstrations is an easy way to see seismic data processing!

1.12 Self Documentation in Seismic Un*x

To display documentation for programs like **suplane** or **suxwigb**, enter the name of the program without arguments. "Without arguments" means you enter only the name of an SU program. For example, enter

```
suplane
```


Downloaded 01/21/14 to 132.178.2.64. Redistribution subject to SEG license or copyright; see Terms of Use at http://library.seg.org/

SU interprets this to mean you want information about **suplane**, so SU prints the program help (selfdoc) file to the screen. Below are the first several lines of the **suplane** selfdoc file.

```
SUPLANE - create common offset data file with up to 3 planes

suplane [optional parameters] >stdout

Optional Parameters:
 npl=3                  number of planes
 nt=64                  number of time samples
 ntr=32                 number of traces
 taper=0                no end-of-plane taper
                        = 1 taper planes to zero at the end
 offset=400             offset
 dt=0.004               time sample interval in seconds
```

You see that the name stands for "create common offset data file with up to 3 planes." (Many SU program names start with "su.") You also see the default values (values used when the user does not supply a value) of various parameters, such as the number of planes (*npl=3*), the number of time samples per trace (*nt=64*) and the time sample interval (*dt=4 ms*). Using default values, each trace is 256 ms long.

1.13 Your Second Seismic Un*x Instruction

Let's change the command given earlier in Section 1.8. Enter:

```
suplane  npl=2  dt=0.008  |  suxwigb  &
```

By supplying values for *npl* and *dt*, you overwrite (replace) **suplane**'s default values. Your output (pictured below) has only two planes and each trace is 512 ms long.

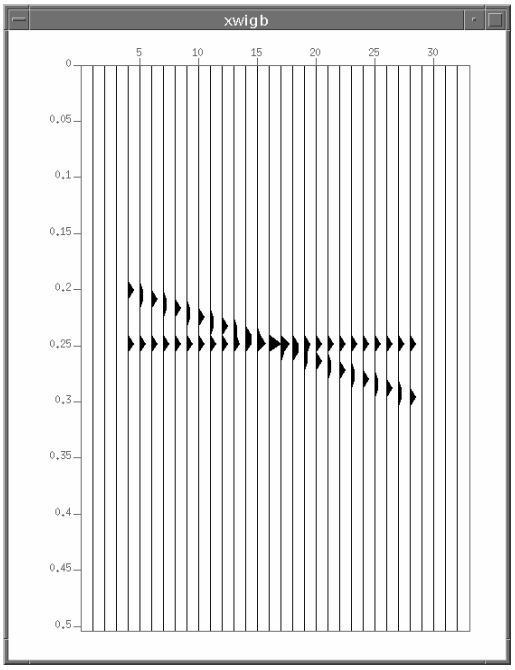

Figure 1.7: *The **suplane** command with two defaults overwritten: number of planes = 2, and time interval = 0.008 seconds.*

Downloaded 01/21/14 to 132.178.2.64. Redistribution subject to SEG license or copyright; see Terms of Use at http://library.seg.org/

1.13.1 Computer Note 6: No Spaces around the Equal Sign

Do not use spaces before or after the equal (=) sign

*1.14 Other Help Facilities in Seismic Un*x*

Various help facilities are described in The New SU User's Manual by John W. Stockwell, Jr. & Jack K. Cohen; Version 3.2: August 2002. This Manual is available at the CWP Seismic Un*x web site. In addition to the self documentation described in Section 1.11, other help facilities include:

- **suhelp** lists all the available programs.
- **suname** lists all programs and libraries with a short description of them.
- **sudoc** followed by the program name gives documentation of the program. This can work even when there is no selfdoc for the program.
- **sufind** followed by a string searches in all self documentation for this string.
- **sukeyword** lists keys used for headers.
- **demos** is a directory in the SU installation that contains useful scripts.

Downloaded 01/21/14 to 132.178.2.64. Redistribution subject to SEG license or copyright; see Terms of Use at http://library.seg.org/

Downloaded 01/21/14 to 132.178.2.64. Redistribution subject to SEG license or copyright; see Terms of Use at http://library.seg.org/